AD-A271 820

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

(2)

DATE	3. REPORT TYPE AND DATES COVERED
	FINAL/01 SEP 89 TO 30 NOV 93

4. TITLE AND SUBTITLE
ALGORITHM/ARCHITECTURE STUDY FOR ARTIFICIAL NEURAL NETS (U)

5. FUNDING NUMBERS
2305/B3
AFOSR-89-0501

6. AUTHOR(S)
Professor Sun-Yuan Kung

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Princeton University
Department of Electrical Eng
Princeton, NJ 08544-0036

8. PERFORMING ORGANIZATION REPORT NUMBER
AFOSR-TR- 93 0611

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NM
110 DUNCAN AVE, SUITE B115
BOLLING AFB DC 20332-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFOSR-89-0501

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED

12b. DISTRIBUTION CODE
UL

13. ABSTRACT (Maximum 200 words)

Neural information processing has already helped catalyzed many potential opportunities of cross-fertilization, from which many diversified disciplines have benefited mutually. However, in order to substain a long-term impact, there must establish a fundamental and coherent theory for it. This project focuses on the development and understanding of the fundamental system theoretical basis for temporal dynamic networks. The main thrust of the research hinges on a thorough understanding of several key issues regarding temporal dynamical system modeling, including model unification, training efficiency, generalization performance, and hierarchical network structure.

14. SUBJECT TERMS

15. NUMBER OF PAGES
18

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR(SAME AS REPORT)

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements***.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	**PR**	- Project
G	- Grant	**TA**	- Task
PE	- Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- **DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- **DOE** - See authorities.
- **NASA** - See Handbook NHB 2200.2.
- **NTIS** - Leave blank.

Block 12b. Distribution Code.

- **DOD** - Leave blank.
- **DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- **NASA** - Leave blank.
- **NTIS** - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

FINAL REPORT ON GRANT AFOSR-89-0501

Algorithm/Architecture Study for Artificial Neural Nets

S.Y. Kung, Princeton University

1 Overview of Research Accomplishments

Neural information processing has already helped catalyzed many potential opportunities of cross-fertilization, from which many diversified disciplines have benefited mutually. However, in order to substain a long-term impact, there must establish a fundamental and coherent theory for it. This project focuses on the development and understanding of the fundamental system theoretical basis for temporal dynamic networks. The main thrust of the research hinges upon a thorough understanding of several key issues regarding temporal dynamical system modeling, including model unification, training efficiency, generalization performance, and hierarchical network structure.

- **Distributed Training Strategy and Decision-Based Neural Net**

  We have studied a class of neural models based on distributed credit-assignments, with hard or *fuzzy* decision rules, leading to theoretical understanding of multi-modal analysis and a structural unification of neural models.

- **Temporal Dynamic Models**

  The aim is to design models which best capture the transient characteristics and/or contextual information of temporal patterns. We shall

explore a prediction-based neural classifiers and compared it with other temporal models such as HMM, TDNN and RNN. More fundamentally, we shall pursue a unification framework for temporal dynamic models, based on a Hamiltonian-Jacobian system theory. A new energy formulation will be introduced to tackle the issues of training efficiency and generalizability of temporal networks.

- **Principal Component Analysis and Applications**

  We have developed a new learning network model for Principal Component of a single process and extended the learning models for extracting the asymmetric principal components of a pair of signals. The lateral connection network enforces orthogonality between the feed-forward weights which is essential for the extraction of the orthogonal principal components. The novel structure also facilitates the growing or shrinking of the network whenever an order update is required.

- **Generalization**

  The focus of study is placed on identification and capacity theoretical perspectives of learning and generalization; numerical analysis for training efficiency, and network reduction and growing techniques for improving generalizability.

- **Digital Parallel Neural Processor**

  A unified model supporting various connectivity structures and nonlinear functions is developed. This leads to the design of neural array processors based on highly pipelined ring systolic/wavefront architectures. Specifically, one-dimensional and two-dimensional array structures for various neural networks have been developed. In addition, we have developed

a high-level timing simulations by *SISim*. A precise simulation tool is required in order to obtain an estimate closer to the real world.

Overally, we have completed the following objectives:

- Neural networks for static and temporal pattern recognitions and a unified formulation for both application domains.
- Develop versatile neural models, including a decision-based neural net (DBNN), and a fuzzy-decision neural net (DBNN), and study the generalization performances.
- Experimental study and comparison based on real-world application examples.
- Stress numerical analysis regarding learning rates and convergence properties to facilitate digital implementations.

In summary, these research works represent an array of related and comprehensive research tasks which are outstanding but yet very critical to a coherent treatment fo neural models. For temporal pattern recognition, it will be beneficial to explore the rich theoretical relationship between the novel neural classifiers and the conventional theory on system identification and nonlinear filter. Ultimately, the theoretical analyses should be in one way or another realized in real-world applications on temporal pattern recognition. In a broader sense, the study naturally covers the algorithmic bases for distributed and massively parallel processing, so it will have a very positive implication to the future on-line information processing technologies. The detailed technical discussion can be found in a recent book by Kung [5]. Soe sample cases of the aforementioned accomplishments will be discussed below.

2 Distributed Training Strategy and Decision-Based Neural Net

2.1 Theoretical Study

Supervised networks may be systematically developed according to several key design factors, including training strategy, temporal property, network structure, and training criterion. The training formulation can be either decision-based or back-propagation approximation-based or hybrid. Key research issues such as distributed and localized credit assignments and hierarchical system design are yet to be thoroughly investigated. To focus on training only critical subnets or subnodes, a novel *localized credit assignment* scheme is adopted in the decision-based neural network (DBNN) [5]. To improve generalizability, proper cost criterion (with a "soft" decision) may be incorporated. This leads to the development of the so-called fuzzy-decision neural networks (FDNN) [8]. The DBNNs and FDNN can be naturally applied to temporal recognition problems due to its simple winner-take-all principle. In fact, the independent training is conceptually and computationally very appealing to temporal models. The DBNN structure may be combined with temporal discriminant functions including, for examples, DTW, prediction error, and likelihood function.

1. **Distributed Training Strategies**

   The training strategy may be greatly influenced by the credit-assignment schemes used. For best efficacy, the design hierarchy is purposefully divided into two stages. The first involves individual training by achieving an optimally trained model function $\phi(\cdot)$. Its performance may be further improved by incorporating mutual training methods used in the DBNN.

The individual training method (either discriminative or intrinsic) is often formulated under an optimization formulation. For the *independent training* strategy, each subnet is trained by the positive-examples only and it implies potential computational saving. Note however that the training can be effective only when the cost criterion is properly chosen. Following the independent training phase, the mutual training phase will be executed when there is a need to further enhance the overall performance. This scheme is particularly suitable to temporal pattern recognitions. We shall further explore the two-phase training strategy so as to envision the optimal hybrid scheme combining the merits of both independent and mutual training.

2. Fuzzy Decision Neural Networks

The objective is to improve generalization performance. This may be accomplished by incorporating vigilance or tolerance into the mutual training strategy. A more formal approach is via a notion of fuzzy-decision neural networks(FDNN)[5, 8]. In the FDNN, a penalty function as a function of the degree of errors should be adopted. Since a linear cost function would impose an unproportional penalty for the patterns with extremely large error, an inverse exponential function is more appealing. It effectively treats the errors with equal penalty once the the magnitude of error exceed certain threshold, making the cost criterion very much in accordance with the so-called "minimum-error-rate" criterion by Duda [3]. In addition, the new format also facilitates a network to produce multiple options each assigned a corresponding probability, instead of one single best decision. This precipitates the use of fuzzy rules in the subsequent information processing. We shall also explore the theoretical

justification on convergence and generalization as well as the selection of proper penalty functions.

2.2 Applications of Decision-Based Neural Networks

In our research, a new class of decision-based neural networks (DBNN) have been proposed. These networks combine the perceptron-like learning rule with a hierarchical nonlinear network structure, so they are termed HiPer Nets. Two HiPer net structures are proposed: hidden-node and subcluster structures. We shall explore several variants of HiPer nets based on the different hierarchical structures and basis functions and then examine the relationships between HiPer nets and other DBNNs, e.g. Perceptron and LVQ. Based on the simulation performance comparison, the HiPer nets appear to be very effective for many signal/image classification applications, including texture classification, OCR, and ECG.

For more flexibility in the nonlinearity of decision boundaries, two hierarchical structures, i.e. hidden-node and subcluster structures, are considered. In training a complex hierarchical network, the key questions to be addressed are *which* subnets or subnodes to update, and *how* to update. The answer to " *which*" lies in a novel competition-based "credit-assignment" principle: the anti-reinforced learning should be applied to (the winning subnode in) the winning subnet; while the reinforced learning be applied to (the local-winner in) the correct class. (By this mechanism, the weight updating can be most effective and confined to only a subset of "growing" modules.) The answer to " *how*" is provided in Algorithm 1.

Decision-Based Neural Networks

Algorithm 1 (Decision-Based Learning Rule)

Suppose that $S = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(M)}\}$ *is a set of given training patterns, with each pattern* $\mathbf{x}^{(m)} \in R^N$ *belonging to one of the* L *classes* $\{\Omega_i, i = 1, \cdots, L\}$*; and that the transfer functions are* $\phi(\mathbf{x}, \mathbf{w}_i)$ *for* $i = 1, \ldots, L$*. Suppose that the* m*-th training pattern* $\mathbf{x}^{(m)}$ *presented is known to belong to class* Ω_i*; and that the winning class for the pattern is denoted by an integer* j*, i.e. for all* $l \neq j$*,*

$$\phi(\mathbf{x}^{(m)}, \mathbf{w}_j^{(m)}) > \phi(\mathbf{x}^{(m)}, \mathbf{w}_l^{(m)}) \tag{1}$$

(1) *When* $j = i$*, then the pattern* $\mathbf{x}^{(m)}$ *is already correctly classified, so no update will be needed.*

(2) *When* $j \neq i$*, i.e.* $\mathbf{x}^{(m)}$ *is still misclassified, then the following update will be performed:*

$$\begin{aligned} &\textit{Reinforced Learning:} && \mathbf{w}_i^{(m+1)} = \mathbf{w}_i^{(m)} + \eta \nabla \phi(\mathbf{x}, \mathbf{w}_i) \\ &\textit{Anti-Reinforced Learning:} && \mathbf{w}_j^{(m+1)} = \mathbf{w}_j^{(m)} - \eta \nabla \phi(\mathbf{x}, \mathbf{w}_j) \end{aligned} \tag{2}$$

In this learning rule, the reinforced learning moves w along the positive gradient direction, so the value of transfer function will increase, enhancing the chance of the pattern's future selection. The anti-reinforced learning moves w along the negative gradient direction, so the value of transfer function will decrease, suppressing the chance of its future selection.

Radial Basis Function These models use a radial-basis function (RBF) and are very effective for practical applications, especially for nearest-neighbor-type classifications. In this case, a simplest radial-basis transfer function

$$\phi(\mathbf{x}, \mathbf{w}_l) = -\frac{\| \mathbf{x} - \mathbf{w}_l \|^2}{2} \tag{3}$$

is used for each subnet l. Thus the following learning rules can be derived:

$$\begin{aligned} &\text{Reinforced Learning:} && \mathbf{w}_i^{(m+1)} = \mathbf{w}_i^{(m)} + \eta(\mathbf{x} - \mathbf{w}_i^{(m)}) \\ &\text{Anti-Reinforced Learning:} && \mathbf{w}_j^{(m+1)} = \mathbf{w}_j^{(m)} - \eta(\mathbf{x} - \mathbf{w}_j^{(m)}) \end{aligned} \tag{4}$$

This is the basic formula for a Generalized LVQ (GLVQ) algorithm, which is very close to the LVQ2 algorithm.

Elliptic Basis Function In terms of second-order basis functions, the most general form is the (skewed) hyper-elliptic basis function. For simplicity, however, in most application experiments, the EBF is confined to the normal (upright) version.

Hierarchical Network Structure So far we have adopted a nonlinear transfer function represented by a single-layer model in each subnet. However, the single-layer model may be inadequate for very complex decision boundaries. So we resort to a structural solution. Two basic structures of the Hierarchical Perceptron (HiPer) nets are are called *hidden-node structure* and *subcluster structure*, based respectively on a distribution-oriented and a winner-take-all approach. In order to have a consistent indexing scheme for the hierarchical structure, we shall label the subnet level by the index l, and label the subnode level (within a subnet) by the index k_l. In a sense, the HiPer Net learning rule represents a unified framework for a insightful understanding of several prominent decision-based networks, including Perceptron, LVQ, PNN.

Hidden-Node Structure One way to create a more versatile transfer function is to use a two-layer model for each subnet. In this case, a hidden layer is introduced which consists of multiple "hidden" subnodes, each of which represented by a basis function $\psi_l(\mathbf{x}, \mathbf{w}_{k_l})$. The transfer function of the subnet

is a linear combination of the subnode values:

$$\phi(\mathbf{x}, \mathbf{w}_l) = \sum_{k_l=1}^{K_l} c_{k_l} \psi_l(\mathbf{x}, \mathbf{w}_{k_l}) \tag{5}$$

where $\{c_{k_l}\}$ denotes the coefficients in the upper layer and $\mathbf{w}_l$ is the vector comprising all the weight parameters. (We stress the fact that, under the decision-based formulation, there is no need to use a nonlinear unit at the output layer, since it will not affect the classification results.)

The most common basis functions, $\psi_l(\mathbf{x}, \mathbf{w}_{k_l})$, for the subnodes include linear-basis function (LBF), radial-basis function (RBF), and elliptic-basis function (EBF).

Subcluster Structure For the subcluster hierarchical structure, we introduce notions of *local winner* and *global winner*. The *local winner* is the winner among the subnodes within the same subnet. The local winner of the l-th subnet is indexed by s_l, i.e.

$$s_l = Arg \max_{s_l} \psi_l(\mathbf{x}, \mathbf{w}_{s_l})$$

The *global winner* is the winner among all the subnets. The j-th subnet will be labeled as the global winner, if its local winner wins over all the other local winners, i.e.

$$\psi_j(\mathbf{x}, \mathbf{w}_{s_j}) > \psi_l(\mathbf{x}, \mathbf{w}_{s_l}) \quad \forall l \neq j$$

Key Variants of HiPer Nets As listed in Table 1, examples for hidden-node and subcluster HiPer nets are respectively HiPer(L_h), HiPer(R_h), and HiPer(L_s), HiPer(R_s), and HiPer(E_s).

HiPer Nets	transfer function	remark
$HiPer(L_s)$	linear basis	generalization of linear perceptron
$HiPer(R_s)$	radial basis	· also named GLVQ
$HiPer(E_s)$	elliptic	· similar to LVQ
$HiPer(L_h)$	weighted sum of sigmoid of linear basis function	· can use BP algorithm
$HiPer(R_h)$	weighted sum of Gaussian on RBF	· similar to PNN

Table 1: Key variant of HiPer Nets. Here capital letters "L", "R", and "E" stand for linear, radial, or elliptic basis functions respectively. A subscript "s" denotes a subcluster structure, while a subscript "h" denotes a hidden-node structure.

Applications to Signal/Image Classifications In order to test the performance of the DBNNs, several application examples are studied, including texture classification and OCR.

Texture Classification Based on the texture classification applications, we compare the performance of radial-basis and elliptic-basis GLVQs, denoted as HiPer(R_s) and HiPer(E_s) respectively. As a comparison, we have also included a linear-basis hidden-node structure HiPer(L_H) into the study.

The texture feature used here is based on a compressed representation of the *texture spectrum*. The texture vector associated with a pixel is characterized by 8 "ternary" values, {0, 1, or 2}, labeling the relative level between the central pixel and its 8 immediate neighbors. In the simulation study, a total of 12 Brodatz textures (texture numbers 3, 16, 28, 33, 34, 49, 57, 68, 77, 84, 93, and 103) are used. For each texture image, 529 32×32 blocks are sampled uniformly across the entire image. Their reduced spectra are then computed which will in turn used as the training data. By a similar method, additional 200 blocks are randomly chosen from the same texture image to form the test set. The linear-basis hidden-node structure HiPer(L_h), and two subcluster structures: HiPer(R_s) and HiPer(E_s) have been tried. The classification performance is summarized in Table 2. The results indicate a good convergence and accuracy. The generalization performance of HiPer(E_s) is slightly better than that of HiPer(R_s), with the HiPer(L_h) as a distant third.

Optical Character Recognition (OCR) Application The problem is to recognize a rectangular pixel display as one of the 26 capital letters in the English alphabet. The character images were based on 20 different fonts and each letter within these 20 fonts was randomly distorted to produce a file

Network	Noise Tolerance	Test Error Rate
HiPer(L_h)(20)	0(200 sweeps)	3.25%
HiPer(R_s)(4)	0(20 sweeps)	2.88%
HiPer(R_s)(4)	0.13	2.42%
HiPer(R_s)(8)	0(7 sweeps)	3.54%
HiPer(R_s)(8)	0.2	2.92%
HiPer(E_s)(1)	0(200 sweeps)	3.04%
HiPer(E_s)(4)	0(17 sweeps)	2.46%
HiPer(E_s)(4)	0.15	2.08%

Table 2: Comparison of various HiPer nets for the texture classification. The number in the parenthesis denotes the number of subnodes in the subnet. The classification rates on the training set are 100% for all the models. The exceptions are 99.5 for HiPer(E_s)(1) and 98.4 % for HiPer(L_h)(20). We have also observed that, with some additional adjustment of learning rate, the HiPer(E_s)(4) can achieve as low as 1.4 % in the error rate.

Algorithm	No. of Clusters	Training Error	Test Error
HiPer(E_s)	10	0.0375%	6.2%
HiPer(R_s)	20	0.025%	7.5%

Table 3: Comparison of two HiPer nets for the OCR classification.

of 20,000 stimuli. Some sample characters are used in the OCR classification experiments. Several variations of Holland-style adaptive classifier systems was previously investigated by by Frey (1991) and the best accuracy obtained was a little over 80%. We have used the first 16000 items as the training patterns to training the network. Then in the testing phase, for the remaining 4000 are used as the testing patterns, for which the trained network is used to predict the letter category.

Two HiPer nets are tried for this application and the simulation results are summarized in Table 3. In this simulation, 10-subcluster $HPN(E_s)$ performs better than 20-subcluster $HPN(R_s)$. (Note the number of the weight parameters are approximately the same for the two nets.) Both HiPer nets perform better than the results previously reported.

ECG Signal Classification In general, we are inclined to state that the DBNNs appear to be superior to the approximation-based nets in terms of both the convergence speed and training accuracy. [1] According to our experimental study on ECG classifications, cf. Table 4, they seem to retain the edge in the generalization performance as well.

[1] Indeed, the texture experiments also showed that the DBNNs out-perform the (BP) approximation-based nets. The BP method was very slow and the mean-square-error remained very large after 500 sweeps. So the experiments were stopped.

	Algorithm	Test Accuracy
Approximation-based	LBF-BP(20)(mse:0.05)	78%
	RBF-OCON(4) (mes:0.1)	80%
Decision-based	HiPer(E_s)(2)	90%
	HiPer(R_s)(4)	90%

Table 4: Comparison of ECG classification for 10 signal classes. The training accuracies are 100% for all the models. OCON stands for one-class-one-net. For HiPer(E_s)(2), a noise tolerance of 0.5 is adopted.

3 Temporal Dynamic Models

3.1 Theoretical Study

There is an emerging need of a fundamental approach to the modeling and analysis of temporal networks. For temporal pattern recognition, the transient characteristics and contextual information of the signals must be accounted for. Therefore, the temporal dynamic modeling deserves to be looked at from an innovative perspective. The conventional TDNN suffers from several critical drawbacks in terms of real-world applications. Its complexity usually incurs time-consuming training process. The prefixed span of time-delay often renders it less suitable for heavily warped (speech) signals. Accordingly, the issue of selecting an optimal structure of temporal dynamic model (TDM) remains a very open topic. In the following, we propose several theoretical and empirical approaches to the study of temporal behaviors.

1. **Temporal Model Understanding**

For temporal pattern recognition, it is critical to incorporate memory units into the neural network, e.g. time-delay units in deterministic networks or Markovian state transition in stochastic networks. The questions are when and which one to use for the best possible cost/performance efficiency. We shall examine how to cope with the tradeoff between the improved recognition performance and higher network complexity in terms of a large number of memory units. We note that, for example, some structures are naturally suitable for differentiation or integration of the signals. Thus the temporal dynamic network should be designed according to the application needs. Among several competing techniques are recurrent nets and Markov models. There are significant distinctions in the fundamental properties between the two prominent models. Our study also shows that they are very much related, theoretical and application ground exists for possible cross-fertilization [5]. We propose to develop a hybrid model which integrates the strengths of BP, RNN, and HMM. A successful integration would imply smaller network size, better temporal robustness/generalizability, and faster training efficiency.

2. Recognition of Transient and Other Temporal Characteristics

The question now is how to design networks to best capture the transient characteristics of temporal patterns. The proposed model is the *prediction-based independent training*(PBIT) network, based on the same principle as linear predictive filter [10, 5]. Its training involves *positive-examples* only. A Gaussian network defined as a linear combination of N-dimensional Gaussians, denoted by $N(x, \mu, \Sigma)$, can be used as a universal approximator or predictor. The network can capture the more eventful segments in the training waveforms. Therefore, the classifier

can be tolerant to the shifting and warping of the signals. Moreover, the transient characteristics can be easily captured by the predictive error criterion. To determine the optimal filter-order (window size) of PBIT, a generalization or information criterion may be adopted. For a relative performance study, the nonrecurrent prediction-based models will be compared with other recurrent models, e.g. [11].

The nonlinear predictive filter has potentially other applications. In the recursive identification problem, the recorded output from the system is compared to that of an adjustable model. The model parameters are updated according to the difference until the difference cannot be further improved. A similar approach can be applied to the adaptive control procedure which compares the actual output of the plant with that of a reference model, and make adjustments in the regulator until the plant output coincides with the model output.

3. Unification of Temporal Dynamic Models

Many theoretical techniques for training recurrent neural networks are proposed based on a generalized BP learning rule, in which the gradients are computed via backpropagation through both the time and the space. Based on a Hamiltonian-Jacobian framework, a new energy formulation may be introduced to tackle the very important issues of training efficiency and generalizability of temporal networks. It leads to an intriguing interplay between the time for dynamic behavior t versus the training time s the system parameters. With an extra set of derivative variables, it is possible to compute the gradient without the explicit "time" backpropagation. Another issue is the stability of nonlinear neural systems [1]. Because the RBF models have already made a

major inroad in temporal pattern recognition, it is important to extend the present Hamiltonian TDM analysis to the RBF models. In a even broader setting, such temporal dynamic models could provide a unified framework between LBF/RBF neural models, adaptive nonlinear filters, stack filters, and IIR (infinite-impulse-response) filters [4, 7].

3.2 Applications of Temporal Networks

Prediction-Based Independent Training Networks(PBIT) The PBIT is suitable exclusively for temporal pattern recognition. The theoretical basis of the PBIT follows that of the linear predictive filter, which is very popular in the signal processing research community. The main distinction of PBIT from the linear predictive classifier lies in its use of nonlinear neural functions.

In training a PBIT networks, an $\bar{N}$-dimensional training signal x is first put through a time-delay neural network, cf. Figure 1, where many time-delayed N-dimensional vector segments can be extracted:

$$\mathbf{x}_j = [x(j+1)\ldots x(j+N)]$$

where $x(j)$ denotes the j_{th} element of the pattern x. A Gaussian network (see Figure 1), defined as a linear combination of N-dimensional Gaussians, denoted by $N(x, \mu, \Sigma)$, can be used as a universal approximator or predictor. To predict $x(j+N+1)$ by $\mathbf{x}_j$, let us use the following predictor:

$$f_a(\mathbf{x}_j) = \sum_{k=1}^{K} w_k N(\mathbf{x}_j, \mu_k, \Sigma_k) + w_0$$

where the corariance matirx Σ_k is a diagonal matrix. For each class we assign a Gaussian network to it and the prediction error for a pattern (say, m-th

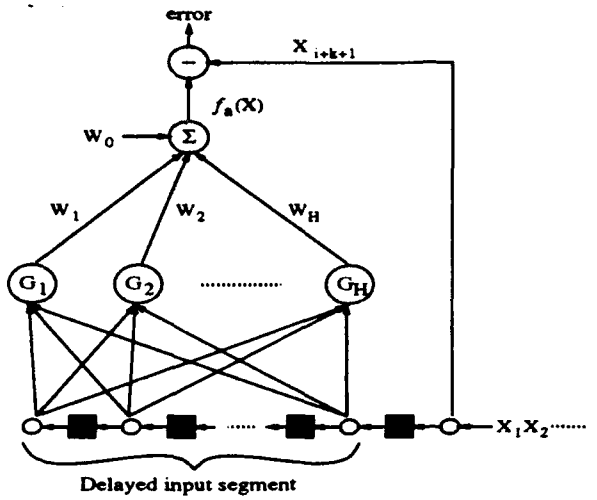

Figure 1: A Gaussian RBF TDNN for prediction of a future sample.

pattern) is defined as

$$E^{(m)} = \sum_{j=1}^{\hat{N}-N} (f_a(\mathbf{x}_j) - x(j+N+1))^2$$

By this formulation, the classifier is made more tolerant to the shift and the length of the signal. The centroids of the Gaussian function can help capture the information about the key segments in the or iginal signal.

Here we adopt an independent training strategy. For each class, a Gaussian network described above is trained so that the sum of the prediction errors from all the training patterns, $m = 1, \cdots$,M:

$$E = \sum_{m=1}^{M} E^{(m)}$$

is minimized.

In the retrieving phase, given a test pattern $\mathbf{x}^{(m)}$, the prediction errors $E^{(m)}$ for all the classes will be computed and compared. The pattern x_m is classified into the class with the smallest prediction error.

Network	Training Set	Test Set
PBIT	100%	96%
HMM	100%	92%
DBNN(E_s)	100%	90%

Table 5: In this simulation, the window size for PBIT is 20 and the variances of the Gaussians are fixed to be 1.

Simulation Results: ECG Classification of PBIT To demonstrate the feasibility of PBIT, some simulation results are reported here. In this simulation, we apply PBIT to 10 ECG classes. Each class has 10 patterns and half of them are used as training set. The segment from the sliding window and the next sample value are adjusted by the mean value of the segment so that the classifier is more tolerant to local DC level of the ECG signal. The simulation results are summarized in Table 5. PBIT yields the same training performance, as compared with the HMM (also an independent training method) and the DBNN (a mutually training model). In terms of the generalization accuracy, PBIT compares favorably with the HMM and DBNN.

HMM for Time-Warped Signal Classification To test the power of the HMM, some time-rescaled ECG waveforms are used for recognition. The original ECG data are composed of 10 classes with 10 waveform patterns in each class. Those ECG patterns were resampled with respect to different sampling rates ($50/T$, $60/T$, and $70/T$ in this experiment where T is the period of a single ECG pulse). The resampled data were then quantized into 20 levels which correspond to 20 observation symbols. The HMM is independently trained.

Sampling Rate	First Group	Second Group
$50/T$	100%	88%
$60/T$	100%*	90%
$70/T$	98%	92%

Table 6: The test accuracies with the respect to different groups of resampled data. As marked by the "*", this test set is equivalent to the training set, i.e. the training accuracy is 100 %.

For each sampling rate, there are 100 sampled waveforms. These are further divided into two groups with 50 waveforms each. The first group of $60/T$ sampled waveforms is used as the training set, while the remaining groups are used as the testing sets. The testing accuracies are displayed in Table 6. The results indicate that the HMM does exhibit a good tolerance for the time-rescaled waveforms.

4 Principal Component Analysis and Applications

A connection is made between APEX and the Recursive Least Squares (RLS) algorithm which provides us with an optimal value for the step-size parameter and drastically improves the performance of the algorithm. We have shown that the convergence rate of the network to be exponential and we are in fact able to analytically approximate it for each component while we verify our prediction via simulation.

We have justified the feasibility of a parallel APEX model, which we then simulate and study its convergence properties. Furthermore, two learning net-

work models based on the Hebbian rule are proposed for extracting the Principal Oriented Component of a pair of signals. Mathematical analysis based on approximation assumptions proves the asymptotic convergence of the first model to the desired component.

We have proved that in a two-layer supervised linear feed-forward network with fewer hidden units than input and output units the optimal solution to the least-squares criterion is related to the Generalized Singular Value Decomposition between the input data matrix and the input-output outer product matrix. This holds true even if the input data are rank-deficient, i.e. the input autocorrelation matrix is non-singular. We call the problem Linear Approximation Asymmetric PCA since PCA is a special case when the teacher of the network is equal to the input of the network. Wiener filtering is also shown to be a special case of linear approximation APCA.

We have verified that the least-squares cost function contains no local minima under reasonable assumptions and therefore, the gradient descent type algorithms like Back-Propagation will be able to achieve the global minimum. However the minimum is not unique and BP extracts some linear combination of the asymmetric PCA components rather than the components themselves. In order to extract the exact generalized singular components related to linear approximation APCA we propose to use a lateral connection network among the hidden units in addition to the feed-forward connections that exist in the standard BP network. We propose two types of learning rules for the lateral net: the dynamic and the local orthogonalization rules.

The cross-correlation APCA problem is shown to be related to the SVD of the cross-correlation matrix of two signals and can be tackled by another proposed linear learning network. The network again features the lateral con-

nections we found useful in both the PCA and the linear approximation APCA problems. It is f mally shown that our model indeed extracts the desired singular vectors of the cross-correlation matrix and the accompanying simulations verify this claim and also imply that the convergence is exponential.

5 Generalization

The primary research issues are how to come up with an effective and practical generalization criterion and how it affects the estimation the optimal number of clusters or hidden-units. A promising approach towards this objective is to combines the knowledges from neural information processing, system identification theory and temporal dynamic system analysis. In order to achieve a high computational efficiency, numerical and convergence analyses of neural models are indispensable. It is also important to investigate the estimation of optimal learning rate and convergence speed rate for multilayer networks based on both backpropagation and orthogonal learning rules. The scheme has been successfully applied to the APEX networks [6].

1. **Identification Perspective of Learning and Generalization**

   An important design issue for multi-layer networks is the number of hidden units, which dictates the space separability and the discriminating capability of the network. The optimal hidden-layer size depends on the trade-off between *training accuracy* and the *generalization accuracy*. We shall propose a flexible and systematic scheme in the selection of the energy function for the training phase so as to improve generalizability. For example, by regularization it avoids direct penalty due to an exceeding number of neurons.

The generalizability is the most critical criterion. We shall study from both perspectives of identification (curve-fitting the sample points) and capacity (minimizing risk). Minimizing the following generalization criterion (see [9])

$$E \approx \rho(1 + \frac{d(\lambda)}{M}) \tag{6}$$

leads to one optimal estimate of the size. The formulation also suggests that the marginal effects on the "effective dimension" diminish when the size of the network becomes extraordinarily large. So some weights may be removed without causing too much degradation on the generalization performance. This provides a starting point for a theoretical study on the estimation of the optimal size. In this study, the theoretical analysis will be supplemented by real experiments to evaluate the proper criterion (ML, LSE, etc) for real-world applications.

2. **Training Efficiency: Numerical analysis**

For many real-time applications, fast and parallel updating algorithms are indispensable. The question is how to improve training efficiency and better control its numerical behavior. We have already demonstrated an analytical approach which estimates optimal learning rates. The result tightly match the real numerical simulation results for special PCA nets [5, 2]. For the general nets, we propose to derive the optimal learning rates directly via the *recursive least square* algorithm, with or without a forgetting factor. More advanced, the Jacobian training formulation sheds light on the generalization performance as well as the numerical training efficiency. The Jacobian matrix of a feedforward network with nonlinear sigmoid output neurons is often ill-conditioned. Due to the ill-conditionedness, there are redundant weights in the net-

work. Thus we shall study proper neuron and basis functions to avert the ill-conditionedness and enhance discriminative capability. We will also analyze the eigenvalues of the Jacobian matrix and use this tool to remove redundancy.

6 Digital Parallel Neural Processors

A unified model supporting various connectivity structures and nonlinear functions is developed. This leads to the design of neural array processors based on highly pipelined ring systolic/wavefront architectures. Specifically, one-dimensional and two-dimensional array structures for various neural networks have been developed.[5] In addition, we have developed a high-level timing simulations by *SISim*. A precise simulation tool is required in order to obtain an estimate closer to the real world.

In order to get an accurate information about the behavior of a system, both the hardware and the software must be specified precisely. This motivates the development of a simulation tool - *SISim*. The *SISim* simulator is a system level interactive simulator developed at Princeton. *The most important application of SISim is to help estimate more accurately the total computation time*. There are two input file modules required for *SISim* processing: one for the hardware and the other for the software. The hardware description module contains a 'cfg' file and several 'hwd' files. 'cfg' file specifies the configuration of the target system, and 'hwd' file gives a more detailed description for the hardware of each processor. The software description module, on the other hand, contains a 'prg' file and several 'swd' files. The 'prg' files associate each processor with a program which this processor should execute, and the detailed program is given by a 'swd' file. Figure 2 illustrates a description of a matrix-vector

multiplication on a 5-processor array.

The *SISim* itself consists of three components: *PARSER*, *DRIVER*, and *ISIM*. The *PARSER* reads the hardware and software description modules and creates an internal data structure which will be used as the base of the next (*DRIVER*) simulation stage. The *DRIVER*, based on a time-table, is used to simulate the behavior of the target system. When *SISim* is used in an interactive mode, the *ISIM* will also be invoked, which works together with *PARSER* and *DRIVER*. The *ISIM* allows the user to check the status of any system component at any time during the simulation period. The final statistics concerning the real execution time for each processor is stored in the 'rst' file.

SISim may be applied to the back-propagation network to determine more realistic speed-up-factors for the linear array or the rectangular array implementations. It also provides a more exact analysis of how the different parameters (e.g. computation time, communication time, buffer size, memory fetch, etc.) affect the overall performance of the array processors.

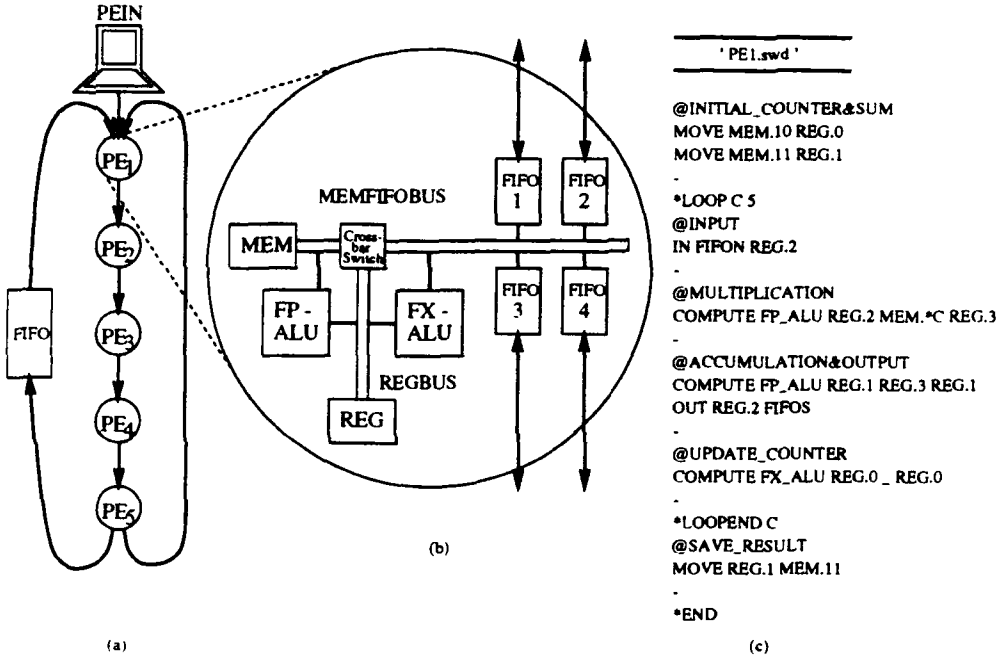

Figure 2: An example for neural net retrieving phase. (a) Configuration description for the whole system. (b) The hardware description for a single processor. (c) The software description for the program executed by a single processor.

References

[1] L. O. Chua and L. Yang. Cellular neural networks: theory and applications. *IEEE Transactions on Circuits and Systems*, 35(10):1257–1290, 1988.

[2] K. Diamantaras and S. Y. Kung. Multi-layer neural networks for reduced-rank approximation. *In press, IEEE Transactions on Neural Networks*, 1993.

[3] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* John Wiley, New York, 1973.

[4] B. H. Juang, S. Y. Kung, and C. A. Kamm (Editors). *Neural Networks for Signal Processing.* Proceedings of the 1991 IEEE Workshop, Princeton, NJ, 1991.

[5] S. Y. Kung. *Digital Neural Networks.* Prentice Hall, Englewood Cliffs, NJ, 1993.

[6] S. Y. Kung and K. I. Diamantaras. A neural network learning algorithm for adaptive principal component extraction (APEX). In *Proceedings, IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 861–864, Albuquerque, NM, April 1990.

[7] S. Y. Kung, F. Fallside, J. A. Sorensen, and C. A. Kamm (Editors). *Neural Networks for Signal Processing, II.* Proceedings of the 1992 IEEE Workshop, Helsingoer, Denmark, 1992.

[8] S. Y. Kung and J. S. Taur. Fuzzy-decision neural networks and some applications. Proceedings, IEEE International Conference on Acoustics, Speech, and Signal Processing, April 1993.

[9] L. Ljung and J. Sjoberg. A system identification perspective on neural nets. In S. Y. Kung, F. Fallside, J. A. Sorensen, and C. A. Kamm, editors, *Neural Networks for Signal Processing, II.* Proceedings of the 1992 IEEE Workshop, Helsingoer, Denmark, 1992.

[10] J. S. Taur and S. Y. Kung. Prediction-based networks with ECG application. In *Proceedings, IEEE International Conference on Neural Networks*, San Francisco, March 1993.

[11] N. H. Wulff and J. . A. Hertz. Prediction with recurrent networks. In S. Y. Kung, F. Fallside, J. A. Sorensen, and C. A. Kamm, editors, *Neural Networks for Signal Processing, II*, pages 464–473. Proceedings of the 1992 IEEE Workshop, Helsingoer, Denmark, 1992.

7 List of Publications

1. **Textbook**

   S. Y. Kung, *Digital Neural Networks*, Prentice Hall, Englewood Cliffs, NJ, 1993.

2. **Book-Edited** B.H. Juang, S.Y. Kung, and C.A. Kamm (Editors), *Neural Networks for Signal Processing*, Proceedings of IEEE Workshop, 1991.

3. **Book-Edited** "Neural Networks for Signal Processing, II" Publisher: IEEE Press, 1992. (Editors: S.Y. Kung, F. Fallside, J.A. Sorensen, C. Kamm)

4. "Hidden Markov Models for Character Recognition" IEEE Transactions on Image Processing, volume 1, pp. 539-543, October, 1992, (J. A. Vlontzos and S. Y. Kung)

5. "Multi-layer Neural Networks for Reduced-Rank Approximation", In press, IEEE Transactions on Neural Networks, 1993. (K. Diamantaras and S.Y. Kung)

6. "Decision-based Hierarchical Neural Networks with signal/Image Classification Applications." in press, IEEE Transactions on Neural Networks, 1993.(Kung and Taur)

7. "Adaptive Principal Component EXtraction (APEX) and Applications" in press, IEEE Transactions on Signal Processing, 1993 (S.Y. Kung, K.I. Diamantaras, and J.S. Taur)

8. "Fuzzy-Decision Neural Networks", Proceedings, IEEE International Conference on Acoustics, Speech, and Signal Processing, Minneappolis, April,

1993. (J.S. Taur and S.Y. Kung)

9. "Prediction-Based Networks with ECG Application", Proceedings, IEEE International Conference on Neural Networks, San Francisco, March 1993. (J.S. Taur and S.Y. Kung)

10. "Mapping Neural Networks onto Array Processors" as Chapter 1 of "Digital Implementations of Neural Networks", (Editors: V. Prassana Kumar and W. Przytula), pp. 1- 25, Prentice-Hall, Inc. 1993. (S. Y. Kung and W.H. Chou)

11. "On Partitioning and Fault Tolerance Issues for Neural Array Processors", In press, accepted by Journal on VLSI Signal Processing, (Karl-Heinz Zimmermann, Mike T. C. Lee, and S.Y. Kung)

12. "Parallel and Pipelined VLSI Implementation of a Staged Decoder for BCM Signals". to appear on Journal of VLSI Signal Processing, (G. Caire, J. Ventura-Traveset,J. Murphy, and S.Y. Kung)

13. "Hierarchical Perceptron (HiPer) Networks for Classifications," Proceedings, IEEE Workshop on Neural Networks for Signal Processing, Helsingoer, Denmark, pp. 267-278, 1992. (S.Y. Kung and J.S. Taur)

14. "SISim: A System-level Interactive Simulator for Array Processor System," Proceedings, IEEE Workshop on VLSI Signal Processing, Napa Valley, Ca., Oct 28-30 1992. (W. H. Chou and S. Y. Kung)

15. "VLSI Systolic Array Implementation of a Staged Decoder for BCM Signals" "VLSI Signal Processing, V", (edit by Kung Yao et al.), IEEE Workshop on VLSI Signal Processing, Napa Valley, Ca., Oct 28-30 1992, pp. 139-149, (G. Caire, J. Ventura-Traveset,J. Murphy, and S.Y. Kung)

16. "Array Processor Design System for Neural Networks", Proceedings, International Computer Symposium, pp. 1100- 1122, Taichung, December, 1992. (W. H. Chou and S. Y. Kung)

17. J.S. Taur and S.Y. Kung, Comparison of several learning subspace methods for classification, In *Proceedings, ICASSP91*, May 1991.

18. S.Y. Kung and J.S. Taur, Hierarchical Perceptron (hiper) networks for classifications, In *Proc. NNSP92*, 1992.

19. S.Y. Kung and W.H. Chou, Mapping Neural Networks onto VLSI Array Processors, In K.W. Przytula and V.K. Prasanna, editors, *Digital Parallel Implementations of Neural Networks*. Prentice-Hall, 1992.

20. S. Y. Kung and Yu Hen Hu, A frobenius approximation reduction method (farm) for determining optimal number of hidden units, In *Proceedings of the IJCNN-91*. IJCNN, Seattle, Washington, July 1991.

21. W. H. Chou, Sisim: A system level interactive simulator, Technical Report, Computer Engineering Group, Dept. of E. E., Princeton University, 1991.

22. W.H. Chou and S.Y. Kung, SISim: A System-level Interactive Simulator for Array Processor System, In *Proceedings of IEEE Workshop on VLSI Signal Processing*, 1992.

23. K. I. Diamantaras and S. Y. Kung, "Multilayer Neural Networks for Reduced-Rank Approximation under Rank-Deficient Input Data," to appear in *IEEE Transactions on Neural Networks*.

24. S. Y. Kung, K. I. Diamantaras and J. S. Taur, "Neural Networks for Extracting Pure / Constrained / Oriented Principal Components," in *SVD and Signal Processing, II: Algorithms, Analysis and Applications*, R. Vaccaro ed., pp. 57-81, Elsevier, 1991.

25. S. Y. Kung, K. I. Diamantaras, W. D. Mao and J. S. Taur, "Generalized Perceptron Networks with Nonlinear Discriminant Functions," in *Neural Networks, Theory and Applications.*, R. J. Mammone and Y. Zeevi eds., pp. 245-279, Academic Press, 1991.

26. S. Y. Kung and K. I. Diamantaras, "Neural Networks for Extracting Unsymmetric Principal Components," in *Neural Networks for Signal Processing - Proceedings of the 1991 IEEE-SP Workshop*, pp. 50-59, Princeton, NJ, September 1991.

27. K. I. Diamantaras and S. Y. Kung, "An Unsupervised Neural Model for Oriented Principal Component Extraction," in *Proceedings Int. Conf. Acoustics Speech and Signal Processing (ICASSP-91)*, pp. 1049-1052, Toronto, Canada, May 14-17, 1991.

28. J.N. Hwang, S.Y. Kung, "Parallel Algorithms/Architectures for Artificial Neural Networks," Journal of VLSI Signal Processing, Vol. 1, No. 3, pp. 221-251, 1990.

29. J. A. Vlontzos and S. Y. Kung, "Hidden Markov Models for Character Recognition" IEEE Trans. on Acoustics, Speech, and Signal Processing, Jan. 1990.

30. S. Y. Kung, J. A. Vlontzos, J. N. Hwang, "VLSI Array Processors for Neural Network Simulation," *Journal of Neural Network Computing*,

Auerbach Publishing Co., New York, Vol. 1, No. 4, Spring 1990.

31. S.Y. Kung, "Adaptive Principal Component Analysis via Orthogonal Learning Network" Proceedings, ISCAS, New Orleans, 1990.

32. S. Y. Kung and K. I. Diamantaras, "A Neural Network Learning Algorithm for Adaptive Principal Component Extraction (APEX)," in Proceedings Int. Conf. Acoustics Speech and Signal Processing (ICASSP-90), pp. 861-864, Albuquerque N.M., April 1990.

33. S.Y. Kung, W.D. Mao, Toynori Ishida, "Parallel processors for real time vision analysis", Proceedings, ISCAS, New Orleans, 1990.

34. Sam Sun, S. Y. Kung, "Approximate String Embedding in a Labeled Graph", Proceedings, IEEE Int. Symp. on Information Theory, Jan. 1990.

35. K. I. Diamantaras, K. H. Zimmermann and S. Y. Kung, "Integrated Fast Implementation of Mathematical Morphology Operations in Image Processing," in Proceedings Int. Symp. Circuits and Systems, pp. 1442-1445, New Orleans, May 1990.

36. K. I. Diamantaras, D. L. Heine and I. D. Scherson, "Implementation of Neural Network Algorithms on the P^3 Parallel Associative Processor," in Proceedings Int. Conf. in Parallel Processing ICPP-90, pp. 247-250, St. Charles IL, August 13-17, 1990.

Journal/Keynote Publications on Neural Nets. Good for AFOSR report

37. "Neural Network Architectures for Robotic Applications," *IEEE Trans. on Robotics and Automation*, special issue on *Computational Algorithms*

and Architectures in Robotics and Automation, Vol. 5, No. 5, pp. 641-657, October, 1989. (Kung and Hwang)

38. "A Unified Systolic Architecture for Artificial Neural Networks," *Journal of Parallel and Distributed Computing*, special issue on *Neural Networks*, 1989. (Kung and Hwang)

39. "A Systolic Neural Network Architecture for Hidden Markov Model," IEEE Trans. on Acoustics, Speech, and Signal Processing, December 1989. (Hwang, Vlontzos and Kung)

40. "VLSI Array Processors for Neural Network Simulation," to appear in *Journal of Neural Network Computing*, Auerbach Publishing Co., New York, Vol. 1, No. 4, Spring 1990. (S. Y. Kung, J. A. Vlontzos, J. N. Hwang)

41. "Parallel Algorithms/Architectures for Artificial Neural Networks," to appear in *Journal of VLSI Signal Processing*, Vol. 1, No. 3, 1990. (J. N. Hwang, S. Y. Kung)

42. "From VLSI Arrays to Neural Networks," **Keynote Paper**, International Symposium on Computer Architecture and DSP, Hong Kong, Oct. 1989. (S. Y. Kung)

8 PH.D. THESES COMPLETED UNDER PARTIAL SUPPORT OF THE GRANT

1. "Algorithms/Applications/Architectures for Artificial Neural Nets" (Dr. James Hwang)

2. "Hidden Markov Models for Character Recognition" (Dr. John Vlontzos)

3. "Character Recognition via Waveform/String Matching: Algorithms and Architectures" (Dr. Sam Sun)

4. "Neural Network Algorithms and VLSI Architectures for Pattern Classification" (Dr. W.D. Mao)

5. "Principal Component Learning Networks and Applications". (Dr. K. I. Diamantaras)

6. "Decision Based Neural Models for Pattern Recognitions" (Dr. J.S. Taur) Expected Graduation: June 1993.

7. "Neural Net for Circuit Routing Applications", (Dr. Mark Goudreau)